-
$$f(x) = e^{-|x|} - \frac{x}{1 + x^2}$$

```
import math
def f(x) :
    return math.exp(-abs(x)) - x/(1 + x * x)
```

- Three ways to compute $x^2$
  - `pow(x,2.) = exp(2. * log(x))`
  - `x**2. = exp(2. * log(x))`
  - `x * x    # fast and preferable`

-
$$f'(x) = \begin{cases} -e^{-x} + \frac{x^2-1}{(1+x^2)^2}, \text{if } x > 0 \\ e^{x} + \frac{x^2-1}{(1+x^2)^2}, \text{if } x < 0 \end{cases}$$

- Consider two programs

```
def fp(x) :
    return (x*x-1)/(1+x*x)/(1+x*x) + \
                -math.exp(-x) if x > 0 else math.exp(x)
```

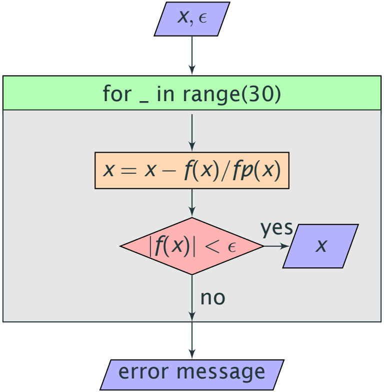
```
def fp(x) :
    x2 = x * x
    return (x2-1)/(1+x2)/(1+x2) + \
                -math.exp(-x) if x > 0 else math.exp(x)
```

- 중간(tempoary) 변수를 잘 활용하여야 한다.

- Newton–Raphson method for $f(x) = 0$
    - Repeat

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, n = 0, 1, \ldots$$

until $|f(x_{n+1})| < \epsilon$ or $n = 30$(say)



```
def newton(x, eps = 0.00001):

    for _ in range(30):
        x = x - f(x)/fp(x)
        if abs(f(x)) < eps :
            break

    else :
        print(”No solution”)
        return ()
    return x
```

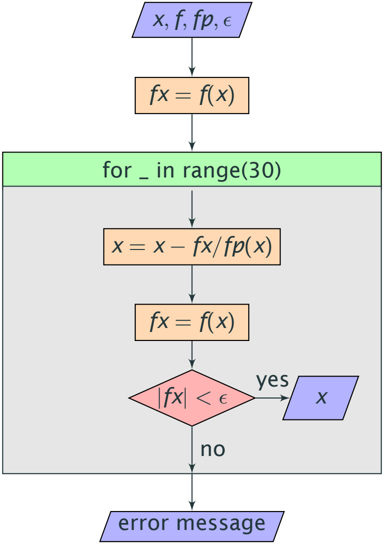
- Three issues:
  - break

```
def newton(x, eps = 0.00001):

    for _ in range(30):
        x = x - f(x)/fp(x)
        if abs(f(x)) < eps :
            return x

    print(”No solution”)
    return ()
```

  - Generalization: function name이 f and fp인 경우만 적용 가능

```
def newton(x, f, fp, eps = 0.00001):
```

  - 계산상의 문제: 불필요한 계산
    - step 0: f(x0), fp(x0), x1 and f(x1) 계산
    - step 1: f(x1), fp(x1), x2 and f(x2) 계산
    - f(x1) 중복 계산


$x, f, fp, \epsilon$
$fx = f(x)$
for _ in range(30)
$x = x - fx/fp(x)$
$fx = f(x)$
$|fx| < \epsilon$
yes
$x$
no
error message